International Journal of Engineering Research
Volume No.5, Issue No.11, pp : 867-870
ISSN:2319-6890)(online),2347-5013(print)
1 Nov. 2016

# Mean-Difference Round Robin (MDRR) Algorithm with Dynamic Time Slice for Task Scheduling Using Real time applications

**[1]A.Gopi,[2]R.Akhilesh Reddy**
[1]Asst.Professor ,CSE,Qis Engg College, City & state?
[2]Asst.Professor,CSE,CMR Institute Of Technology
Email: [1]Gopi.arepalli400@gmail.com,[2]rakhilesh574@gmail.com

***Abstract : Process scheduling means allocating a certain amount of CPU time to each of the user Processes. This paper proposes a CPU Scheduling algorithm, which is meant for optimizing CPU scheduling for real time applications. It is a new approach, which uses the concept of Mean Difference Round Robin (MDRR) with dynamic time quantum, whose value changes for each cycle. It acts better than RR and Mean Difference Round Robin (MDRR) Algorithm in terms of reducing the number of context switches, average waiting time and average turnaround time. The experimental results of the proposed algorithm have been compared with Mean-Difference Round Robin (MDRR) Algorithm is found to have produced optimum scheduling***.
Keywords : ***MDRR, CPU***

## I. Introduction

### Scheduling

Modern Operating Systems are moving towards multitasking environments where CPU scheduling plays a fundamental role, since the CPU is the most effective or essential resource of the computer. CPU scheduling is the process of allocating the CPU to a specific process for a time slice. It is a complex decision making activity because of conflicting goals, limited resources and the difficulty in accurately modelling real world scenarios, and requires careful attention to ensure fairness and avoid process starvation. In multi-programming systems, multiple processes that are being kept in memory for maximum utilization of CPU.CPU utilization can be maximized by switching CPU among waiting processes in the memory and running some process all the time [IV] [V] [VI] [VII].

One of the most difficult problems in designing the operating systems is the timely allocation of resources to the processes and retrieving them. This problem is solved by the classical view of the different algorithms. All of the algorithms have some advantages and disadvantages. However the main aim of the CPU scheduling algorithms is to maximizing CPU utilization and throughput and minimizing turnaround time, waiting time, response time and number of context switching for a set of requests [XII].

### CPU Scheduler

CPU scheduler is a major component of operating systems, Whenever the CPU becomes idle, the operating system must select one of the processes in the queue to be executed. The selection process is carried out by the CPU scheduler (or short-term scheduler). The scheduler selects a process from the processes in memory that are ready to execute and allocates the CPU to that process, while maintaining scheduling policy to obtain user interactivity, throughput, real time responsiveness, and more.

### Scheduling Criteria

Different CPU scheduling algorithms have different properties, and the choice of a particular algorithm may favour one class of processes over another. In choosing which algorithm to use in a particular situation, we must consider the properties of the various algorithms [VIII] [IX]. Many criteria have been suggested for comparing CPU Scheduling algorithms. Which characteristics are used for comparison can make a substantial difference in which algorithm is judged to be best [I] [II] [III]. The criteria include the following:

**CPU utilization, Throughput, Waiting time, Turnaround time, Response time.**

So a good scheduling algorithm for real time and time sharing system are concluded that must possess following characteristics:

- **Maximum CPU utilization,**
- **Maximum throughput,**
- **Minimum turnaround time,**
- **Minimum waiting time,**
- **Minimum response time, and**
- **Minimum context switches.**

### Basic CPU Scheduling Algorithms

For performing the task of CPU scheduling there are various conventional scheduling algorithms. Some of them include: First Come First Serve, Shortest Job First, Round Robin, Priority Scheduling.

### Problem:

The Existing algorithm does not provide the optimal scheduling and it does not provide the better performance of the System.

ISSN:2319-6890)(online),2347-5013(print)
1 Nov. 2016

**Proposed Solution**:
In this paper we reduce the average waiting time and average turnaround time and context switches so finally we have improve the performance of the system and provide optimal scheduling .

## II. Methodology

The proposed algorithm calculates the average burst time of all the processes in the ready queue. Next, it finds out the mean difference i.e. the difference between a process burst time and the calculated average burst time. This step is repeated for all the processes in the ready queue. Then, find out the process having the largest difference value and assigns it to CPU, and execute it for one time slice[X] [XI]. Once the time slice of the process expires, then consider the remaining burst time of running process, if it is equal to or less than the one time slice then continue its execution otherwise the next process with the largest difference value is picked up from the ready queue and executed for one time slice and then check its remaining burst time whether to continue its execution or not [XII]. The process is repeated for all the processes in the ready queue, this is the end of first cycle. Repeat the whole process until all the processes in the ready queue completes their execution. For every cycle, the time quantum is taken as TQ=n*k, where n=1, 2, 3…respectively [I] [II] [III].

## III. Proposed Algorithm

***Input:***
*Bt [] is the array of burst times of all processes in the ready queue.*
*Pid [] is the array of processes identifiers in the ready queue.*
*N is the total number of processes in the ready queue.*
δ *is the time quantum.*
***Output:***
*Average waiting time, AWT, of all processes.*
*Average turnaround time, ATT, of all processes.*
*Number of Context Switches, NCS.*

*Step 1*: Calculate the mean burst time for each process in the ready queue.

Sum =+ Bt [i]; m=sum/N

*Step 2*: Calculate the mean differences for each process in the ready queue

S [] = m-Bt [i];

*Step 3*: Find out the process having the largest difference value and assign it to CPU and execute it for one time slice.

Pid []. Bt[i] = Bt[i]-δ;

*Step 4*: If the remaining burst time of current running process is less than or equal to TQ, continue its execution.

Pid []. Bt [i] <= δ;

*Step 5:* After finishing step 4 pick up the next process having the largest value and execute it and repeat step 4

Find Smax-1 [] and Pid [];

if any two processes have the same mean difference value then find the shortest process and assign it to CPU .

*Step 6*: Repeat step 5 for all the processes in the ready queue.
*Step 7*: For every cycle TQ is taken as δ*k, where k=1, 2, 3...
*Step 8*: Repeat step 1 through step 7 until all the processes in the ready queue are finished.
*Step 9*: Calculate the average waiting time of all the processes in the ready queue.

AWT=Σ Wi/N;

*Step 10*: Calculate the average turnaround time of all the processes in the ready queue.

ATT=Σ Ti/N;

## IV. Results and Discussions

**Comparison of Performance between Existing and New Algorithm**

**Example 1:** Consider the following set of processes, assumed to have arrived at time t0, in the order p1, p2, p3, p4 with the length of the CPU burst given in milliseconds and time quantum is 10, 15, 20, 25 milliseconds.

**Table 1.** Processes with their burst times in milliseconds

| Process Name | Burst Time |
|---|---|
| P1 | 53 |
| P2 | 17 |
| P3 | 68 |
| P4 | 24 |

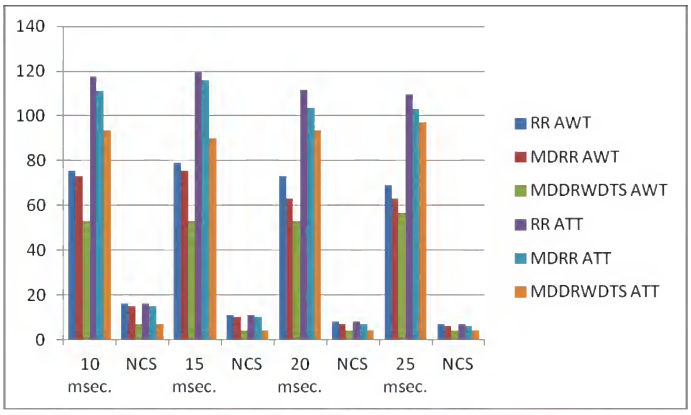
The following table shows the comparative study of the performance of the Mean
Difference Round Robin (MDRR) Algorithm and the proposed Scheduling Algorithm.

**Table 2.** Comparison of the performance of the Mean Difference Round Robin and the proposed Scheduling Algorithm

| Time Slice | 10 msec. | NCS | 15 msec. | NCS | 20 msec. | NCS | 25 msec. |
|---|---|---|---|---|---|---|---|
| RR AWT | 75.5 | 16 | 79.25 | 11 | 73 | 8 | 69.25 |
| MDRR AWT | 73 | 15 | 75.5 | 10 | 63 | 7 | 63 |
| **MDDRWDTS AWT** | **53** | **7** | **53** | **4** | **53** | **4** | **56.75** |
| RR ATT | 117.5 | 16 | 119.5 | 11 | 111.5 | 8 | 109.75 |
| MDRR ATT | 111 | 15 | 116 | 10 | 103.5 | 7 | 103 |
| **MDDRWDTS** | **93.5** | **7** | **89.75** | **4** | **93.5** | **4** | **97.25** |

ISSN:2319-6890)(online),2347-5013(print)
1 Nov. 2016

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ATT** | | | | | | | RR ATT | 156.4 | 21 | 152.8 | 15 | 139.4 | 13 | 137.6 |
| | | | | | | | MDRR ATT | 148.4 | 20 | 152.6 | 16 | 139.4 | 12 | 137.6 |
| | | | | | | | **MDDRWDTS ATT** | **124.4** | **10** | **117.6** | **7** | **119.2** | **7** | **120.8** |




**Graph for Comparison of the performance of the Mean Difference Round Robin and the proposed Scheduling Algorithm**

**Example 2:** consider the following set of processes, assumed to have arrived at time
t0, in the order p1, p2, p3, p4, p5 with the length of the CPU burst given in milliseconds and time quantum is 10, 14, 16, 18 milliseconds.

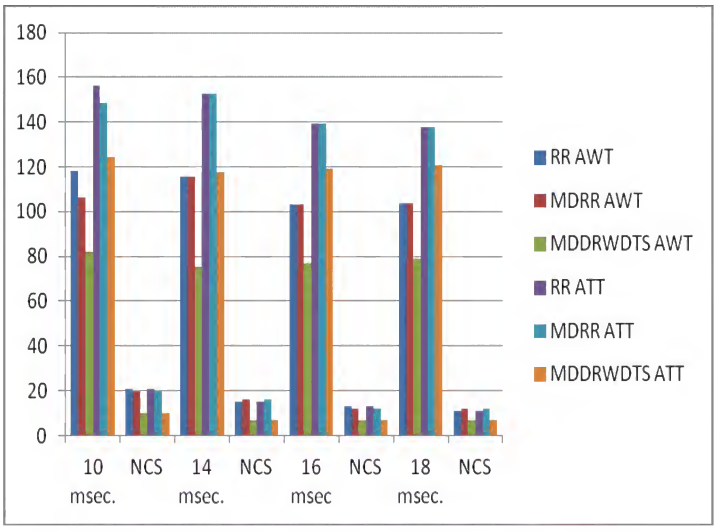
**Table 3.** Processes with their burst times in milliseconds

| Process Name | Burst Time |
|---|---|
| P1 | 48 |
| P2 | 26 |
| P3 | 54 |
| P4 | 16 |
| P5 | 66 |

The following table shows the comparative study of the performance of the Mean Difference Round Robin (MDRR) Algorithm and the proposed Scheduling Algorithm.

**Table 4.** Comparison of the performance of the Mean Difference Round Robin and the proposed Scheduling Algorithm

| **Time Slice** | **10 msec.** | **NCS** | **14 msec.** | **NCS** | **16 msec** | **NCS** | **18 msec.** | **NCS** |
|---|---|---|---|---|---|---|---|---|
| RR AWT | 118.4 | 21 | 115.6 | 15 | 103.2 | 13 | 104 | 11 |
| MDRR AWT | 106.4 | 20 | 115.6 | 16 | 103.2 | 12 | 104 | 12 |
| **MDDRWDTS AWT** | **82.4** | **10** | **75.6** | **7** | **77.2** | **7** | **78.8** | **7** |




**Graph for Comparison of the performance of the Mean Difference Round Robin and the proposed Scheduling Algorithm**

## V.Conclusion

In this paper a CPU Scheduling algorithm is proposed, which is meant for optimizing CPU scheduling for real time applications. The performance of the proposed algorithm with the Mean Difference Round Robin (MDRR) algorithm is presented with the help of two examples, in this paper. The results shows that the proposed scheduling algorithm is always giving better performance than MDRR, It has been found that average waiting time, average Turnaround time and context switches have been reduced drastically when compared with the Mean Difference Round Robin Algorithm. This proposed algorithm can be implemented to improve the performance in the systems and can also be used for packet scheduling in the network applications and for scheduling jobs in embedded systems. The approach Can be further refined using the concept of arrival time.

## References

i. *G Siva Nageswara Rao, An Enhanced Dynamic Round Robin CPU Scheduling Algorithm, International Journal of Applied Engineering Research, ISSN 0973-4562 Volume 9, Number 15 (2014) pp. 3085-3098.*

ii. *G Siva Nageswara Rao, Comparison of Round Robin CPU Scheduling Algorithm with Various Dynamic Time Quantum, International Journal of Applied Engineering Research, ISSN 0973-4562 Volume 9, Number 18 (2014) pp. 4905-4916.*

ISSN:2319-6890)(online),2347-5013(print)
1 Nov. 2016

*iii. G Siva Nageswara Rao, A NEW PROPOSED DYNAMIC DUAL PROCESSOR BASED CPU SCHEDULING ALGORITHM, Journal of Theoretical and Applied Information Technology 20th March 2015. Vol.73 No.2,ISSN: 1992-8645.*

*iv. ilberschatz, A., Galvin, P.B., Gagne, G.: Operating system principles, 7th edn.*

*v. Shibu, K.: Introduction to Embedded Systems. THM (2009)*

*vi. Dhamdhere, D.M.: Operating Systems A concept-based approach. Tata McGraw Hill*

*vii. Sinth, A., Goyal, P., Batra, S.: An optimized Round Robin Scheduling Algorithm for CPU Scheduling. IJCSE 02(07), 2383–2385 (2010) .*

*viii. Yaashuwanth, C., Ramesh, R.: A New Scheduling Algorithm. IJCSIS 6(2) (2009) Optimizing CPU Scheduling for Real Time Applications Using MDRR Algorithm.*

*ix. Noon, A., Kalakecch, A., Kadry, S.: A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average. IJCSI 8(3(1)) (May2011) .*

*x. Matarneh, R.J.: Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now Running Processes. American Journal of Applied Sciences 6(10), 1831–1837 (2009) ISSN 1546-9239.*

*xi. Hiranwal, S., Roy, K.C.: Adaptive Round Robin scheduling using shortest burst approach, based on smart time slice. International Journal of computer Science and Communication 2(2), 219–326 (2011) .*

*xii. Matarneh, R.J.: Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the now Running Processes. American Journal of Applied Sciences 6(10), 1831–1837 (2009)*